CLAIMS

What is claimed is:

1. An integrated development environment, comprising:

a document manager for retrieving source code;

an editor for displaying the retrieved source code and providing a means for a user to edit the retrieved source code;

a parser layer which detects the software language of the retrieved source code and which activates rules and logic applicable to the detected software language; and

a visualizer dynamically linked to the editor for displaying graphical representations of flows within the retrieved source code using the rules and logic activated by the parser, wherein the editor, parser layer and visualizer cooperate such that edits made to the source code using the editor are automatically reflected in the graphical representations of flows displayed by the visualizer and edits made to the graphical representations of flows in the visualizer are automatically reflected in the source code displayed by the editor.

2. The integrated development environment as recited in claim 1, wherein the graphical representations of flows depict data flows.

3. The integrated development environment as recited in claim 1, wherein the graphical representations of flows depict program flows.

4. The integrated development environment as recited in claim 1, wherein the graphical representations of data flows are expandable and collapsible.

5. The integrated development environment as recited in claim 1, wherein the source code is programmed in a data manipulation language.

6. The integrated development environment as recited in claim 1, wherein the document manager retrieves all files related to the source code to be edited.

7. The integrated development environment as recited in claim 1, wherein the document manager comprises a site manager and a connectivity layer for retrieving source code from one or more remote computers.

8. The integrated development environment as recited in claim 7, wherein the document manager comprises a security layer for managing secure connections with the one or more remote computers.

9. The integrated development environment as recited in claim 1, wherein the editor comprises a template manager for allowing preprogrammed segment of source code to be placed within the source code being edited.

10. The integrated development environment as recited in claim 9, wherein the template manager is capable of automatically correct segments of the source code.

11. The integrated development environment as recited in claim 9, wherein the template manager is capable of automatically generating segments of the source code.

12. The integrated development environment as recited in claim 1, further comprising a means for allowing the source code to be executed both locally and remotely.

13. The integrated development environment as recited in claim 12, wherein the parser layer further examines error log files generated by the means for allowing the source code to be executed to determine segments of the source code determined to include errors.

14. The integrated development environment as recited in claim 13, wherein the visualizer cooperates with the parser layer to change the appearance of displayed flows as a function of the source code segments determined to have errors.

15. The integrated development environment as recited in claim 13, wherein the editor cooperates with the parser layer to change the appearance of portions of the displayed source code as a function of the software segments determined to have errors.

16. The integrated development environment as recited in claim 13, further comprising a message manager cooperating with the parser layer for displaying debugging hints as a function of the source code segments determined to have errors.

17. The integrated development environment as recited in claim 16, wherein the message manager allows a user to edit and maintain debugging hints for a variety of different errors.

18. A method for examining software, comprising:

retrieving software comprised of source code that is programmed in one of a plurality of programming languages; and

parsing the software as a function of the programming language of the source code for the purpose of displaying a flow representation of the software.

19. The method as recited in claim 18, wherein the flow representation depicts data flow.

20. The method as recited in claim 18, wherein the flow representation depicts program flow.

21. The method as recited in claim 18, further comprising allowing a user to expand and collapse the flow representation.

22. The method as recited in claim 18, further comprising allowing a user to search a network for the software to be retrieved.

23. The method as recited in claim 18, further comprising displaying to the user retrievable software located in the network.

24. The method as recited in claim 18, further comprising displaying the software using an editor, allowing the user to edit the software and, in connection with the editing, parsing the software in real time to update the displayed flow representations to reflect changes in the software resulting from the editing.

25. The method as recited in claim 24, further comprising allowing the user to edit the displayed flow representations and, in connection with the editing, reflecting edits made to the flow representations in the software displayed by the editor.

26. The method as recited in claim 18, further comprising retrieving an error log file generated as a result of an execution of the software, parsing the error log file to determine segments of code having an error, and updating the flow representations to indicate which code segments are determined to have an error.

27. The method as recited in claim 26, further comprising displaying debugging hints corresponding to the errors determined to be within the code.

28. The method as recited in claim 27, further comprising retrieving the debugging hits by browsing a network.

29. The method as recited in claim 28, further comprising allowing a user to customize the debugging hints.

30. The method as recited in claim 18, further comprising displaying a plurality of code templates and allowing a user to select and insert one or more of the code templates into the software.

31. The method as recited in claim 30, further comprising browsing a network to retrieve the code templates for selection by the user.

32. The method as recited in claim 31, further comprising displaying code within a code template in response to a request by the user.

33. The method as recited in claim 31, further comprising automatically correcting segments of the source code.

34. The method as recited in claim 31, further comprising automatically generating segments of the source code.

35. The method as recited in claim 18, wherein the programming languages comprise data manipulation languages.

36. The method as recited in claim 35, further comprising accessing RDBMS services to access and retrieve software.

37. The method as recited in claim 18, wherein the flow representation comprises individual flow elements representative of designated segments of code.

38. The method as recited in claim 37, further comprising displaying comments and execution statistics associated with each flow element in response to a mouse-over selection of the respective flow element.

39. The method as recited in claim 18, further comprising displaying a list of data inspection functions in response to selection of a dataset icon.

40. The method as recited in claim 18, further comprising displaying a list of data discovery functions in response to selection of a dataset icon.